Claims

1.   A method of determining if a first node in a graph may be combined with a second node
in said graph, the method comprising the steps of
-    Determining whether the output buffer of said first node will comprise data that is
similar to the input texture of said second node,
-    Examining each program line in said second node to determine if it negates the
possibility of combining nodes, and
-    Editing program code to replace any first-node-texture references with a single pixel,
where a first-node-texture reference is a reference in the second node's code to a
texture that would have been created by the first node.

2.   The method of claim 1 wherein the step of examining each program line in said second
node to determine if it negates the possibility of combining nodes comprises the step of
renaming local variables.

3.   The method of claim 1 wherein the step of examining each program line in said second
node to determine if it negates the possibility of combining nodes comprises the step of
renaming textures.

4.   The method of claim 1 wherein the step of examining each program line in said second
node to determine if it negates the possibility of combining nodes, comprises the step of
analyzing each said program line to determine if there are dependant texture references.

5.   The method of claim 1 wherein the step of examining each program line in said second
node to determine if it negates the possibility of combining nodes, comprises the step of
analyzing each said program line to determine if there is a reference to a texture that
depends upon the output of said first node and said reference is located at coordinates
stored in a register.

6.   The method of claim 1 wherein the step of determining whether the output buffer of said
first node will comprise data that is similar to the input texture of said second node,
comprises the step of determining if the image pixels represented by the output of said
first node are the same as the image pixels represented by the input of said second node.

7. A method of determining if a first fragment program may be combined with a second fragment program, where each said program is for processing a single graphic element, the method comprising the steps of

    - Determining whether the output of the first program represents relevant pixels that are the same a pixels represented by the input of the second program;

    - Examining each program line in said second program to determine if it negates the possibility of combining said two programs, and

    - Editing program code to replace at least one texture reference with a register reference.

8. The method of claim 7 wherein the step of examining each program line in said second program to determine if it negates the possibility of combining programs comprises the step of renaming local variables.

9. The method of claim 7 wherein the step of examining each program line in said second program to determine if it negates the possibility of combining programs comprises the step of renaming textures.

10. The method of claim 7 wherein the step of examining each program line in said second program to determine if it negates the possibility of combining programs, comprises the step of determining if there are dependant texture references.

11. The method of claim 7 wherein the step of examining each program line in said second program to determine if it negates the possibility of combining programs, comprises the step of determining if there is a texture reference that is dependant upon an output of said first program and said texture reference is located at coordinates stored in a register.

12. A method of determining whether a first GPU program may be combined with a second GPU program for processing, the method comprising the steps of

    - checking a cache,

    - if the cache has no information regarding the said combination, analyze said programs to determine if combination is possible;

        ▪ if the cache indicates that a combination is possible, retrieve and use result as indicated by said cache;

        ▪ if the cache indicates that a combination is not possible, do not analyze said programs for possibility of combining.

13.   The method of claim 12 wherein analyzing said programs comprises the steps of:

-    comparing the output of the first program with the input of the second program,

14.   The method of claim 12 wherein analyzing said programs comprises the steps of:

-    checking each line in the second program to replace references to the output of the first program.

15.   The method of claim 13 wherein analyzing said programs comprises the steps of:

-    checking each line in the second program to replace references to the output of the first program

16.   The method of claim 12 wherein analyzing said programs comprises the steps of:

-    determining if the resident hardware is able to process the combined programs.

17.   The method of claim 13 wherein analyzing said programs comprises the steps of:

-    determining if the resident hardware is able to process the combined programs

18.   The method of claim 14 wherein analyzing said programs comprises the steps of:

-    determining if the resident hardware is able to process the combined programs

19.   The method of claim 12 comprising the further steps of:

-    if said analysis yields a result that the programs may be combined,

  ▪   combine said programs by concatenating,

  ▪   perform register allocation on the combined programs,

  ▪   place the combined programs in cache

20.   The method of claim 13 comprising the further steps of:

-    if said analysis yields a result that the programs may be combined,

  ▪   combine said programs by concatenating,

  ▪   perform register allocation on the combined programs,

  ▪   place the combined programs in cache.

21.   The method of claim 14 comprising the further steps of:

-    if said analysis yields a result that the programs may be combined,

  ▪   combine said programs by concatenating,

  ▪   perform register allocation on the combined programs,

  ▪   place the combined program in cache.

22.   The method of claim 16 comprising the further steps of:

131

- if said analysis yields a result that the programs may be combined,
  - combine said programs by concatenating,
  - perform register allocation on the combined programs,
  - place the combined programs in cache.

23. The method of claim 12 comprising the further steps of:
- if said analysis yields a result that the programs may not be combined,
  - place the result in cache

24. A method of evaluating whether a first GPU program may be combined with a second GPU program, comprising the steps of:
- evaluating each line in the second GPU program,
- if a line contains a local variable, rename said local variable if its name is the same as the name of another local variable in the first program;
- if a line contains an explicit texture coordinate reference, eliminate said explicit texture coordinate reference from the executable part of the program, if said explicit texture coordinate reference is dependant upon said first program, and replace said eliminated texture reference with a single pixel reference.

25. The method of claim 24 further comprising the step of :
- if a line contains a registered texture reference, abort said combination process, if said registered texture reference is to a texture dependent on said first program.

26. The method of claim 24 further comprising the step of checking a cache to determine if said first GPU program may be combined with said second GPU program.

27. A computer-readable medium having computer executable instructions for performing the method recited in any one of claims 1, 7, 12, or 24.